

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-117764

(43)Date of publication of application : 27.04.2001

(51)Int.Cl.

G06F 9/06

G06F 12/00

(21)Application number : 11-293154

(71)Applicant : MITSUBISHI ELECTRIC CORP

(22)Date of filing : 15.10.1999

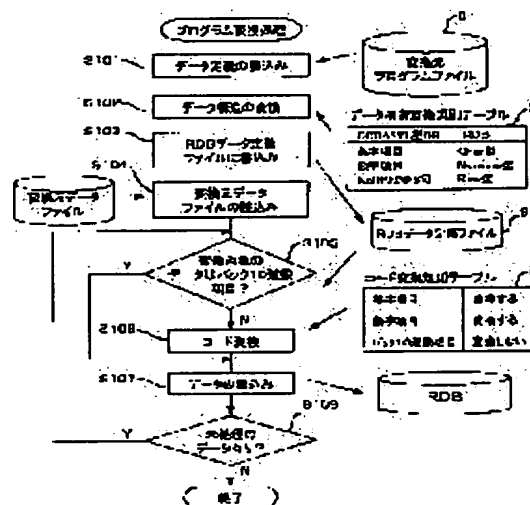
(72)Inventor : HAMAKI HIROTSUGU

(54) METHOD FOR CONVERTING PROGRAM

(57)Abstract:

PROBLEM TO BE SOLVED: To efficiently execute operation for converting a program for accessing an existing data file other than an RDB into a program for accessing the RDB.

SOLUTION: When a pack decimal type data item is defined in a REDEFINES phrase in a program corresponding to a CODASYL type DB described by COBOL language, data items other than the pack decimal type data item out of respective data items included in respective data description terms defined by the REDEFINES phrase are converted into codes when necessary. Then data defined by the REDEFINES phrase are converted into one RAW type data item in each data description term.



LEGAL STATUS

[Date of request for examination]

20.05.2003

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

*** NOTICES ***

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1. This document has been translated by computer. So the translation may not reflect the original precisely.
2. **** shows the word which can not be translated.
3. In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1] The program transformation method characterized by changing the aforementioned program into relational database correspondence by changing into RAW type 1 data item each data description entry defined by the REDEFINES phrase when the packed decimal number type data item is defined in the REDEFINES phrase in the program described in the COBOL language.

[Claim 2] The program transformation method according to claim 1 characterized by not carrying out code conversion to the packed decimal number type data item defined in the REDEFINES phrase when the coding scheme from which the program of a changing agency and the program of a conversion place differ is adopted.

[Translation done.]

* NOTICES *

Japan Patent Office is not responsible for any damages caused by the use of this translation.

- 1.This document has been translated by computer. So the translation may not reflect the original precisely.
- 2.**** shows the word which can not be translated.
- 3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[The technical field to which invention belongs] this invention relates to the method of attaining the increase in efficiency of conversion work in case packed decimal number type data are defined in the method of changing the program using data files other than the relational database (RDB) described in the COBOL language into the program accessed to RDB by shift of a database, especially the REDEFINES phrase.

[0002]

[Description of the Prior Art] Because of downsizing etc., when shifting relatively the program which was being performed on the large-sized host computer to a small personal computer (PC), it is not few. And a database may also be combined and a program may be changed, in case it shifts. For example, in the former, the system using the so-called database management system (DBMS) of the database by the network model based on the way of thinking that between record type may be associated freely, and a CODASYL (The conference on data systems language) method was developed one after another. However, in recent years, in order to express relations, such as attribute value, by the 2-dimensional tabular format, the database by the relational model which can assume each relation easily, and the so-called relational database (RDB) have been making the mainstream. Therefore, in case it downsizes also by the system built by CODASYL type DB, there are not few cases where it shifts to RDB.

[0003] Moreover, at the former, there are not few cases where a spreadsheet and the COBOL language suitable for document creation processing are adopted as a high level language which describes the program which operates on a host computer. For example, the data-definition method in the multi-format peculiar to a COBOL language which is not in other high level languages called a REDEFINES phrase is prepared for the COBOL language. A REDEFINES phrase can be made to access by different data type (category) and merit to the same record storing field if even the length of the whole record which each data description entry defines has agreed. Furthermore, in a COBOL language, the zoned decimal type, the packed decimal number type, and the binary type are prepared as a numerical storing method, and the content to process can define numeric data by the storing method for which were suitable.

[0004] By the way, when it is going to change into the COBOL program of RDB correspondence the COBOL language program which accesses CODASYL type DB, since data type called a Char type and a Number type is prepared, the data item of the character type defined to CODASYL type DB or a binary type can be easily changed into RDB.

[0005]

[Problem(s) to be Solved by the Invention] However, since RDB must define data type and merit beforehand and must form a table, it cannot use the data storage method equivalent to a REDEFINES phrase. Therefore, when the REDEFINES phrase was contained in the program used as the transitional object to PC, two or more data items which decompose each data structure defined by the REDEFINES phrase, and give a definition as a respectively different data item, or constitute each data structure needed to be collectively treated as one data item. For this reason, the data-definition portion (REDEFINES phrase) of the program which was operating on

the host computer had to be changed into RDB correspondence.

[0006] If data type peculiar to a COBOL language called the packed decimal number type showing one number is especially defined by 4 bits, since the data type corresponding to a packed decimal number type does not exist in RDB, When packed decimal number type data are defined in the REDEFINES phrase It could not but store in a position which is different as data of an another section eye as it was shown in drawing 8 (b), though the REDEFINES phrase defined R1 and R2 on the host computer as shown in drawing 8 (a). Therefore, in this case, it had to change with change of a data structure to coding not only in the definition in data division but a procedure division, the repair accompanying the shift work of a program occurred more mostly, and time and effort was great.

[0007] It is made in order that this invention may solve the above problems, and the purpose is in offering the program transformation method that the work which changes the program which accesses the existing data files other than RDB into the program which accesses RDB can be done efficiently.

[0008]

[Means for Solving the Problem] In order to attain the above purposes, the program transformation method concerning this invention changes the aforementioned program into relational database correspondence by changing into RAW type 1 data item each data description entry defined by the REDEFINES phrase, when the packed decimal number type data item is defined in the REDEFINES phrase in the program described in the COBOL language.

[0009] Moreover, when the coding scheme from which the program of a changing agency and the program of a conversion place differ is adopted, code conversion is not carried out to the packed decimal number type data item defined in the REDEFINES phrase.

[0010]

[Embodiments of the Invention] Hereafter, the gestalt of suitable operation of this invention is explained based on a drawing.

[0011] Gestalt 1. drawing 1 of operation is the block diagram having shown the gestalt of the 1 operation of PC which enforces the program transformation method concerning this invention. PC1 has the program transformation processing section 8 which changes a file 6 into the RDB correspondence program file 7 the conversion origin (program) which stores the program described in the COBOL language sent from the host computer 5 by enforcing the program transformation method in the gestalt of this operation based on the code-conversion rule table 2, the data structure transformation-rule table 3, and the RDB table format information file 4.

[0012] Drawing 2 is drawing having shown the example of the data description entry defined using the REDEFINES phrase among the data-definition portions of the COBOL program of program transformation origin. In the COBOL language, although the packed decimal number is expressed with "COMP-3", packed decimal number type data are contained in the data structure defined by the REDEFINES phrase as shown in drawing 2, respectively. Each data is expressed with this example by the EBCDIC code although the example of the data set as the data structure with the REDEFINES phrase shown in drawing 2 is shown in drawing 3.

[0013] Next, the program transformation method which the program transformation processing section 8 in the gestalt of this operation enforces is explained using the flow chart shown in drawing 4.

[0014] The program transformation processing section 8 reads the data division within a program from the changing agency file 6 (Step 101). In addition, the gestalt of this operation explains as a premise that the data definition in the exterior by the INCLUDE sentence etc. is not carried out. When the data definition is carried out outside, also making it applicable to conversion cannot be overemphasized. And the program transformation processing section 8 is changed into the data type in RDB from the data description definition of the head defined as the data division read from the changing agency file 6 at order according to the code-conversion rule set as the code-conversion rule table 2 (Step 102). With the gestalt of this operation, it is characterized by changing into a RAW type the data description definition defined by the REDEFINES phrase. This changed data-definition portion is written in the RDB data-definition file 9 (Step 103).

[0015] With the gestalt of this operation, since not only the program file 6 mentioned above but

the data file (the following, "changing agency data file") which the program file 6 concerned accesses is simultaneously made applicable to conversion, a changing agency data file is read continuously (Step 104). In addition, when two or more changing agency data files exist, about each data file, the respectively same processing is repeated and is performed. Although the program transformation processing section 8 will change each read record data into RDB data, since the 8-bit ShiftJIS (JIS8) code is used for PC1 in the gestalt of this operation, the data expressed with the EBCDIC code adopted by the host computer 5 are changed into JIS8 code. However, it is characterized by the data item of the packed decimal number not considering as the object of code conversion (Step 105,106).

[0016] If this data-conversion processing is explained based on the example shown in drawing 3, since the data key1 "A" of a data name "REC1" shown in drawing 3 are an alphabetic item, they will be changed into JIS8 code. Moreover, since the remaining data data11, data12, and data13 are data of the packed decimal number, code conversion of them is not carried out. The correspondence before and behind this conversion is shown in drawing 5 (a). This code-conversion processing will be carried out according to the code-conversion rule set as the code-conversion rule table 2.

[0017] It processes similarly to the record of a data name "REC2." That is, data22 "01" of the data key2 of an alphabetic item "B" and a numeric item is changed into JIS8 code, respectively. Moreover, code conversion of the data data21 of the remaining packed decimal numbers is not carried out. The correspondence before and behind this conversion is shown in drawing 5 (b). Thus, the data by which code conversion was carried out if needed are written in RDB (Step 107). It is characterized by for the gestalt of this operation defining the record data which carry out code conversion of other data items, and consist of those data items as RAW type 1 data item, without carrying out code conversion of the data item of the packed decimal number among the record data defined by the REDEFINES phrase, and writing in RDB. A database is automatically changed into RDB by performing the above processing to all the records in which it was stored by the changing agency data file.

[0018] The RDB data-definition file 9 generated in the gestalt of this operation is equivalent to an INCLUDE file, if it is based on the example shown in drawing 5, will be read into the RDB correspondence program file 8 as 9 bytes of RAW type data (host variable), and will be transmitted to the object REDEFINES phrase of a DBCOPY phrase. The example of the coding portion of the RDB correspondence program file 8 by which this RAW type data is referred to is shown in drawing 6.

[0019] Drawing 7 (a) is drawing having shown the structure of the record defined by the REDEFINES phrase based on drawing 8 (a) of the conventional example. With the gestalt of this operation, since 1 data structure defined in the REDEFINES phrase is changed as one train, the record of the same structure as the conventional example shown in drawing 7 (a) as shown in drawing 7 (b) is generable. For this reason, even when accessing to the data defined by the REDEFINES phrase in the RDB correspondence program after conversion, it can be used as it is, without being conscious of change of the storing position of each data item which carries out the access etc. On the other hand, if the program transformation method in the gestalt of this operation is not used, it will become record structure as shown in drawing 7 (c). That is, since a data storage position is not in agreement conversion before, the case where the coding portion accessed to the data defined by the REDEFINES phrase must be repaired occurs.

[0020] Since the data structure in the program which suited on the host computer 5 itself has not changed according to the gestalt of this operation, after shifting a program to PC1, it is not necessary to do a data definition again. For this reason, it can be used, without changing the coding portion accessed to the table generated by transform processing mentioned above. Of course, a data-definition portion is also FILE by considering as RDB correspondence to the command about database accesses, such as connect from open, close, read, etc., and select. From SECTION to WORKING-STORAGE Although it must change into SECTION, respectively, since the data definition itself especially the storing position of each data, and length are not changed, the load concerning the part shift work is mitigable.

[0021]

[Effect of the Invention] Since according to this invention it changes so that the data definition of the data containing two or more data items defined using the REDEFINES phrase may be carried out as one RAW type data, the work which changes the program which accesses the existing data files other than RDB into the program which accesses RDB can be done efficiently.

[Translation done.]

*** NOTICES ***

Japan Patent Office is not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.**** shows the word which can not be translated.

3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is the block diagram having shown the gestalt of the 1 operation of PC which enforces the program transformation method concerning this invention.

[Drawing 2] It is drawing having shown the example of a definition of the REDEFINES phrase in the COBOL program of program transformation origin in the gestalt of this operation.

[Drawing 3] It is drawing having shown the example of the data set as the data structure with the REDEFINES phrase shown in drawing 2 .

[Drawing 4] It is the flow chart which showed the program transformation method in the gestalt of this operation.

[Drawing 5] It is drawing having shown the correspondence relation of the record before and behind conversion by the program transformation method in the gestalt of this operation.

[Drawing 6] It is drawing having shown the important section of the program dealing with RDB changed by the program transformation method in the gestalt of this operation.

[Drawing 7] It is drawing having shown the example of the record structure before and behind conversion by the program transformation method in the gestalt of this operation.

[Drawing 8] It is drawing having shown the example of the record structure before and behind conversion by the conventional program transformation method.

[Description of Notations]

1 A personal computer (PC), 2 A code-conversion rule table, 3 A data structure transformation-rule table, 4 A RDB table format information file, 5 A host computer, 6 A changing agency file, 7 A RDB correspondence program file, 8 Program transformation processing section.

[Translation done.]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号
特開2001-117764
(P2001-117764A)

(43) 公開日 平成13年4月27日 (2001.4.27)

(51) Int.Cl. ⁷	識別記号	F I	ターコード [*] (参考)
G 0 6 F 9/06	5 4 0	G 0 6 F 9/06	5 4 0 G 5 B 0 7 6
12/00	5 1 3	12/00	5 1 3 D 5 B 0 8 2

審査請求 未請求 請求項の数 2 O L (全 7 頁)

(21) 出願番号 特願平11-293154

(22) 出願日 平成11年10月15日 (1999. 10. 15)

(71) 出願人 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72) 発明者 濱木 浩継

東京都千代田区丸の内二丁目2番3号 三

菱電機株式会社内

(74) 代理人 100075258

弁理士 吉田 研二 (外2名)

Fターム(参考) 5B076 EA19

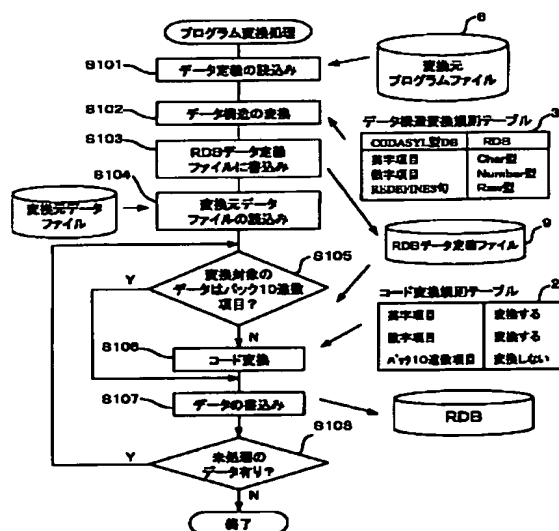
5B082 CA08

(54) 【発明の名称】 プログラム変換方法

(57) 【要約】

【課題】 RDB以外の既存のデータファイルにアクセスするプログラムを、RDBにアクセスするプログラムへ変換する作業を効率的に行わせる。

【解決手段】 COBOL言語で記述されたCODASYL型DB対応のプログラムにおいてバック10進数型のデータ項目がREDEFINES句の中に定義されている場合、そのREDEFINES句により定義された各データ記述項に含まれている各データ項目のうちバック10進数型以外のデータ項目を必要に応じてコード変換する。その後、REDEFINES句により定義されたデータをデータ記述項単位にRAW型の1つのデータ項目に変換する。



【特許請求の範囲】

【請求項1】 COBOL言語で記述されたプログラムにおいてバック10進数型のデータ項目がREDEFINES句の中に定義されている場合、そのREDEFINES句により定義された各データ記述項をRAW型の1データ項目に変換することによって、前記プログラムをリレーショナルデータベース対応へ変換することとを特徴とするプログラム変換方法。

【請求項2】 変換元のプログラムと変換先のプログラムとが異なるコード体系を採用している場合、REDEFINES句の中に定義されているバック10進数型のデータ項目に対してはコード変換をしないことを特徴とする請求項1記載のプログラム変換方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、COBOL言語で記述されたリレーショナルデータベース(RDB)以外のデータファイルを利用したプログラムを、データベースの移行によってRDBへアクセスするプログラムに変換する方法、特にREDEFINES句の中にバック10進数型のデータが定義されている場合の変換作業の効率化を図る方法に関する。

【0002】

【従来の技術】ダウンサイジング等のため、大型のホスト計算機上で実行させていたプログラムを相対的に小さいパーソナルコンピュータ(PC)へ移行する場合は少なくない。そして、プログラムを移行する際、データベースも併せて変更する場合がある。例えば、従来では、レコードタイプの間を自由に関連付けてよいという発想に基づくネットワークモデルによるデータベース、いわゆるCODASYL(The conference on data systems language)方式のデータベースマネジメントシステム(DBMS)を利用したシステムが続々と開発されていた。しかし、近年では、属性値等の関係を二次元の表形式で表現するため各関係を容易に想定できる関係モデルによるデータベース、いわゆるリレーショナルデータベース(RDB)が主流をなしてきている。従って、CODASYL型DBで構築されたシステムでもダウンサイジングを行う際には、RDBへ移行する場合が少なくない。

【0003】また、従来では、ホスト計算機上で動作するプログラムを記述する高級言語としては、表計算、帳票作成処理に適したCOBOL言語が採用される場合が少なくない。例えば、COBOL言語には、REDEFINES句という他の高級言語にないCOBOL言語特有のマルチフォーマットでのデータ定義方法が用意されている。REDEFINES句は、各データ記述項で定義するレコード全体の長ささえ合致していれば、同一レコード格納領域に対して異なるデータ型(項類)・長でアクセスさせることができる。更に、COBOL言語で

は、数値の格納方式としてゾーン10進数型、バック10進数型、バイナリ型が用意されており、処理する内容によって数字データを適した格納方式で定義することができる。

【0004】ところで、CODASYL型DBをアクセスするCOBOL言語プログラムをRDB対応のCOBOLプログラムへ変換しようとする場合、RDBには、Char型、Number型というデータ型が用意されているので、CODASYL型DBに対して定義した文字型やバイナリ型のデータ項目を容易に変換することができる。

【0005】

【発明が解決しようとする課題】しかしながら、RDBは、データ型・長を予め定義してテーブルを形成しておかなければならないので、REDEFINES句に相当するデータ格納方式を利用することはできない。従って、PCへの移行対象となるプログラムにREDEFINES句が含まれている場合、REDEFINES句によって定義された各データ構造を分解してそれぞれ別のデータ項目として定義するか、各データ構造を構成する複数のデータ項目をまとめて1つのデータ項目として扱う必要があった。このため、ホスト計算機上で動作していたプログラムのデータ定義部分(REDEFINES句)をRDB対応へ変更しなければならなかった。

【0006】特に、4ビットで1つの数字を表すバック10進数型というCOBOL言語特有のデータ型が定義されていると、RDBにはバック10進数型に対応するデータ型が存在しないため、REDEFINES句の中にバック10進数型のデータが定義されていた場合には、ホスト計算機上で図8(a)に示したようにR1とR2とをREDEFINES句で定義していたとしても図8(b)に示したように別項目のデータとして異なる位置に格納せざるを得なかった。従って、この場合には、データ構造の変更に伴い、データ部における定義のみならず手続部におけるコーディングまで変更しなければならず、プログラムの移行作業に伴う改修がより多く発生してしまい手間が多大となっていた。

【0007】本発明は以上のような問題を解決するためになされたものであり、その目的は、RDB以外の既存のデータファイルにアクセスするプログラムを、RDBにアクセスするプログラムへ変換する作業を効率的に行うことのできるプログラム変換方法を提供することにある。

【0008】

【課題を解決するための手段】以上のような目的を達成するために、本発明に係るプログラム変換方法は、COBOL言語で記述されたプログラムにおいてバック10進数型のデータ項目がREDEFINES句の中に定義されている場合、そのREDEFINES句により定義された各データ記述項をRAW型の1データ項目に変換

することによって、前記プログラムをリレーショナルデータベース対応へ変換する。

【0009】また、変換元のプログラムと変換先のプログラムとが異なるコード体系を採用している場合、REDEFINES句の中に定義されているバック10進数型のデータ項目に対してはコード変換をしない。

【0010】

【発明の実施の形態】以下、図面に基づいて、本発明の好適な実施の形態について説明する。

【0011】実施の形態1. 図1は、本発明に係るプログラム変換方法を実施するPCの一実施の形態を示した構成図である。PC1は、コード変換規則テーブル2、データ構造変換規則テーブル3及びRDBテーブル構成情報ファイル4に基づき本実施の形態におけるプログラム変換方法を実施することによってホスト計算機5から送られてきたCOBOL言語で記述されたプログラムを格納する変換元(プログラム)ファイル6をRDB対応プログラムファイル7に変換するプログラム変換処理部8を有している。

【0012】図2は、プログラム変換元のCOBOLプログラムのデータ定義部分のうちREDEFINES句を用いて定義されたデータ記述項の例を示した図である。COBOL言語では、“COMP-3”でバック10進数を表すが、図2に示したようにREDEFINES句により定義されたデータ構造には、それぞれバック10進数型のデータが含まれている。図3には、図2に示したREDEFINES句によるデータ構造に設定されたデータの例が示されているが、この例では、各データはEBCDICコードにより表されている。

【0013】次に、本実施の形態におけるプログラム変換処理部8が実施するプログラム変換方法を図4に示したフローチャートを用いて説明する。

【0014】プログラム変換処理部8は、変換元ファイル6からプログラム内のデータ部を読み込む(ステップ101)。なお、本実施の形態では、INCLUDE文等による外部でのデータ定義はされていないことを前提として説明する。もし、外部においてデータ定義がされている場合は、それも変換対象とすることは言うまでもない。そして、プログラム変換処理部8は、コード変換規則テーブル2に設定されたコード変換規則に従い、変換元ファイル6から読み込んだデータ部に定義された先頭のデータ記述定義から順にRDBにおけるデータ型へ変換していく(ステップ102)。本実施の形態では、REDEFINES句で定義されたデータ記述定義をRAW型へ変換することの特徴としている。この変換したデータ定義部分をRDBデータ定義ファイル9に書き込む(ステップ103)。

【0015】本実施の形態では、前述したプログラムファイル6のみならず、当該プログラムファイル6がアクセスするデータファイル(以下、「変換元データファイ

ル」)も同時に変換対象としているので、続いて変換元データファイルを読み込む(ステップ104)。なお、変換元データファイルが複数存在する場合は、各データファイルについてそれぞれ同じ処理を繰り返す。プログラム変換処理部8は、読み込んだ各レコードデータをRDBデータへ変換することになるが、本実施の形態におけるPC1は、8ビットShiftJIS(JIS8)コードを採用しているため、ホスト計算機5で採用されていたEBCDICコードで表されていたデータはJIS8コードに変換される。但し、バック10進数のデータ項目はコード変換の対象としないことを特徴としている(ステップ105、106)。

【0016】このデータ変換処理について図3に示した例に基づき説明すると、図3に示したデータ名“REC1”のデータkey1“A”は、英字項目なのでJIS8コードに変換される。また、残りのデータdata11,data12,data13はバック10進数のデータなのでコード変換されない。この変換前後の対応を図5(a)に示す。このコード変換処理は、コード変換規則テーブル2に設定されたコード変換規則に従い実施されることになる。

【0017】データ名“REC2”のレコードに対しても同様に処理する。すなわち、英字項目のデータkey2“B”、数字項目のdata22“01”は、それぞれJIS8コードに変換される。また、残りのバック10進数のデータdata21はコード変換されない。この変換前後の対応を図5(b)に示す。このように、必要に応じてコード変換されたデータは、RDBへ書き込まれる(ステップ107)。本実施の形態では、REDEFINES句で定義されたレコードデータのうちバック10進数のデータ項目をコード変換せずに、また他のデータ項目をコード変換し、かつそれらのデータ項目で構成されるレコードデータをRAW型の1データ項目として定義してRDBへ書き込むことを特徴としている。以上の処理を変換元データファイルに格納された全レコードに対して行うことでデータベースをRDBへ自動変換する。

【0018】本実施の形態において生成されるRDBデータ定義ファイル9は、INCLUDEファイルに相当し、図5に示した例に基づく、9バイトのRAW型データ(ホスト変数)としてRDB対応プログラムファイル8に読み込み、DBCOPY句の対象REDEFINES句へ転送する。このRAW型データが参照されるRDB対応プログラムファイル8のコーディング部分の例を図6に示す。

【0019】図7(a)は、従来例の図8(a)に基づくREDEFINES句により定義されたレコードの構造を示した図である。本実施の形態では、REDEFINES句において定義した1データ構造を1列として変換するので、図7(b)に示したように図7(a)に示した従来例と同様の構造のレコードを生成することができる。このため、変換後のRDB対応プログラムにおい

てREDEFINES句により定義されていたデータに対してアクセスする場合でもそのアクセスをする各データ項目の格納位置の変更等を意識せずにそのまま使用することができる。一方、本実施の形態におけるプログラム変換方法を利用しないと、図7(c)に示したようなレコード構造となる。すなわち、データ格納位置が変換前と一致しないので、REDEFINES句により定義されていたデータに対してアクセスをするコーディング部分を改修しなければならない場合が発生する。

【0020】本実施の形態によれば、ホスト計算機5上10にあったプログラムにおけるデータ構造自体は変わっていないので、PC1へプログラムを移行した後においてもデータ定義をし直す必要がない。このため、前述した変換処理により生成したテーブルに対してアクセスするコーディング部分を変更することなく使用することができる。もちろん、RDB対応とすることで、open、close、read等からconnect、select等データベースアクセスに関するコマンドへ、データ定義部分もFILE SECTIONからWORKING-STORAGE SECTIONへそれぞれ変更20をしなければならないが、データ定義自体、特に各データの格納位置、長さは変更されないで、その分移行作業にかかる負荷を軽減することができる。

【0021】

【発明の効果】本発明によれば、REDEFINES句を用いて定義された複数のデータ項目を含むデータをRAW型の1データとしてデータ定義するように変換するので、RDB以外の既存のデータファイルにアクセスす*

*るプログラムをRDBにアクセスするプログラムへ変換する作業を効率的に行うことができる。

【図面の簡単な説明】

【図1】 本発明に係るプログラム変換方法を実施するPCの一実施の形態を示した構成図である。

【図2】 本実施の形態においてプログラム変換元のCOBOLプログラムにおけるREDEFINES句の定義例を示した図である。

【図3】 図2に示したREDEFINES句によるデータ構造に設定されたデータの例を示した図である。

【図4】 本実施の形態におけるプログラム変換方法を示したフローチャートである。

【図5】 本実施の形態におけるプログラム変換方法による変換前後のレコードの対応関係を示した図である。

【図6】 本実施の形態におけるプログラム変換方法により変換されたRDB対応のプログラムの要部を示した図である。

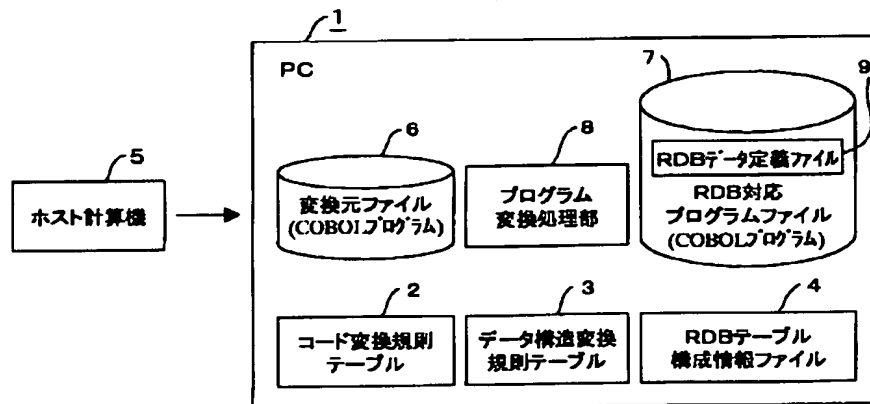
【図7】 本実施の形態におけるプログラム変換方法による変換前後のレコード構造の例を示した図である。

【図8】 従来のプログラム変換方法による変換前後のレコード構造の例を示した図である。

【符号の説明】

1 パーソナルコンピュータ(PC)、2 コード変換規則テーブル、3 データ構造変換規則テーブル、4 RDBテーブル構成情報ファイル、5 ホスト計算機、6 変換元ファイル(COBOLプログラム)、7 RDBデータ定義ファイル、8 プログラム変換処理部、9 RDB対応プログラムファイル(COBOLプログラム)

【図1】



【図2】

```

01  REC 1.
    03  key1    PIC    X.
    03  data11  PIC    S9(003) COMP-3.
    03  data12  PIC    S9(007) COMP-3.
    03  data13  PIC    S9(003) COMP-3.
01  REC 2 REDEFINES REC1.
    03  key2    PIC    X.
    03  data21  PIC    S9(11)  COMP-3.
    03  data22  PIC    XX.
  
```

【図3】

REC1のデータ設定例

```

Key1  = 'A'
data11 = 123
data12 = 1234567
data13 = 123
  
```

C1	12	3C	12	34	56	7C	12	3C
----	----	----	----	----	----	----	----	----

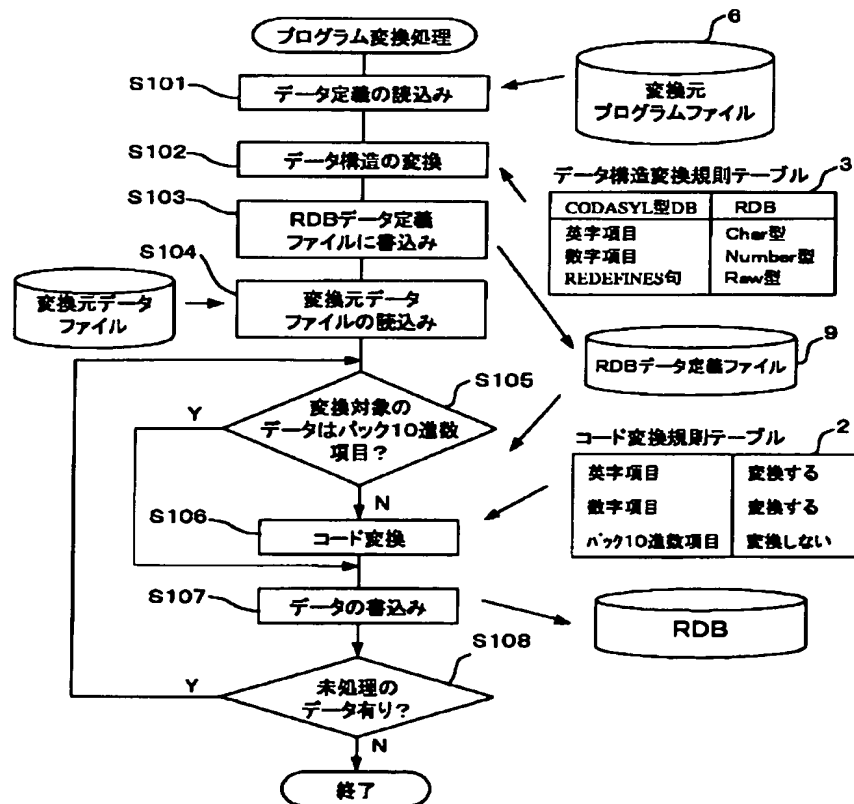
REC2のデータ設定例

```

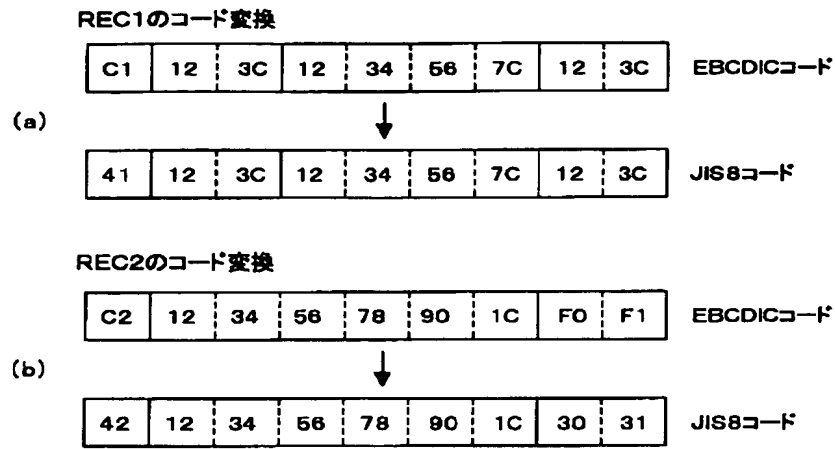
Key2  = 'B'
data21 = 12345678901
data22 = '01'
  
```

C2	12	34	56	78	90	1C	F0	F1
----	----	----	----	----	----	----	----	----

【図4】



【図5】



【図6】

(ホスト変数定義: INCLUDEファイル)

```

...
06 REC PIC X(9).
...
EXEC SQL VAR REC IS RAW(9) END--EXEC.
...

```

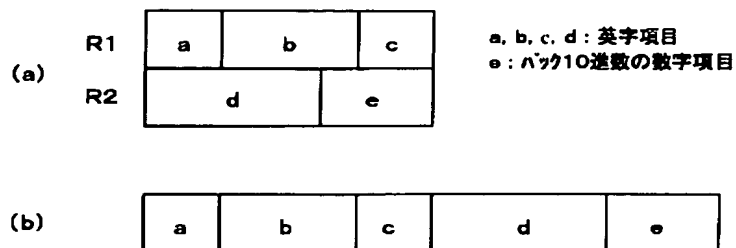
(プログラム)

```

...
EXEC SQL
SELECT *
INTO
...
: REC
...
...
MOVE REC TO REC1.

```

【図8】



【図7】

変換元ファイル

(a)

レコードNo.	キー	REDEFINES項目		
1	R1	a ₁	b ₁	c ₁
2	R2	d ₂		e ₂
3	R2	d ₃		e ₃
4	R1	a ₄	b ₄	c ₄
⋮				



RDBテーブル

(b)

レコードNo.	キー	RAW型		
1	R1	a ₁	b ₁	c ₁
2	R2	d ₂		e ₂
3	R2	d ₃		e ₃
4	R1	a ₄	b ₄	c ₄
		⋮		

列

従来の変換方法によるRDBテーブル

(c)

レコードNo.	キー					
1	R1	a ₁	b ₁	c ₁		
2	R2				d ₂	e ₂
3	R2				d ₃	e ₃
4	R1	a ₄	b ₄	c ₄		

列 列 列 列 列